TrigSim Data Flow

Raw Data Chunk

Copy and Add to Raw Data Chunk

UDC/Direct access

L1 Processors

MBT Channels

L2 Pre-Processors
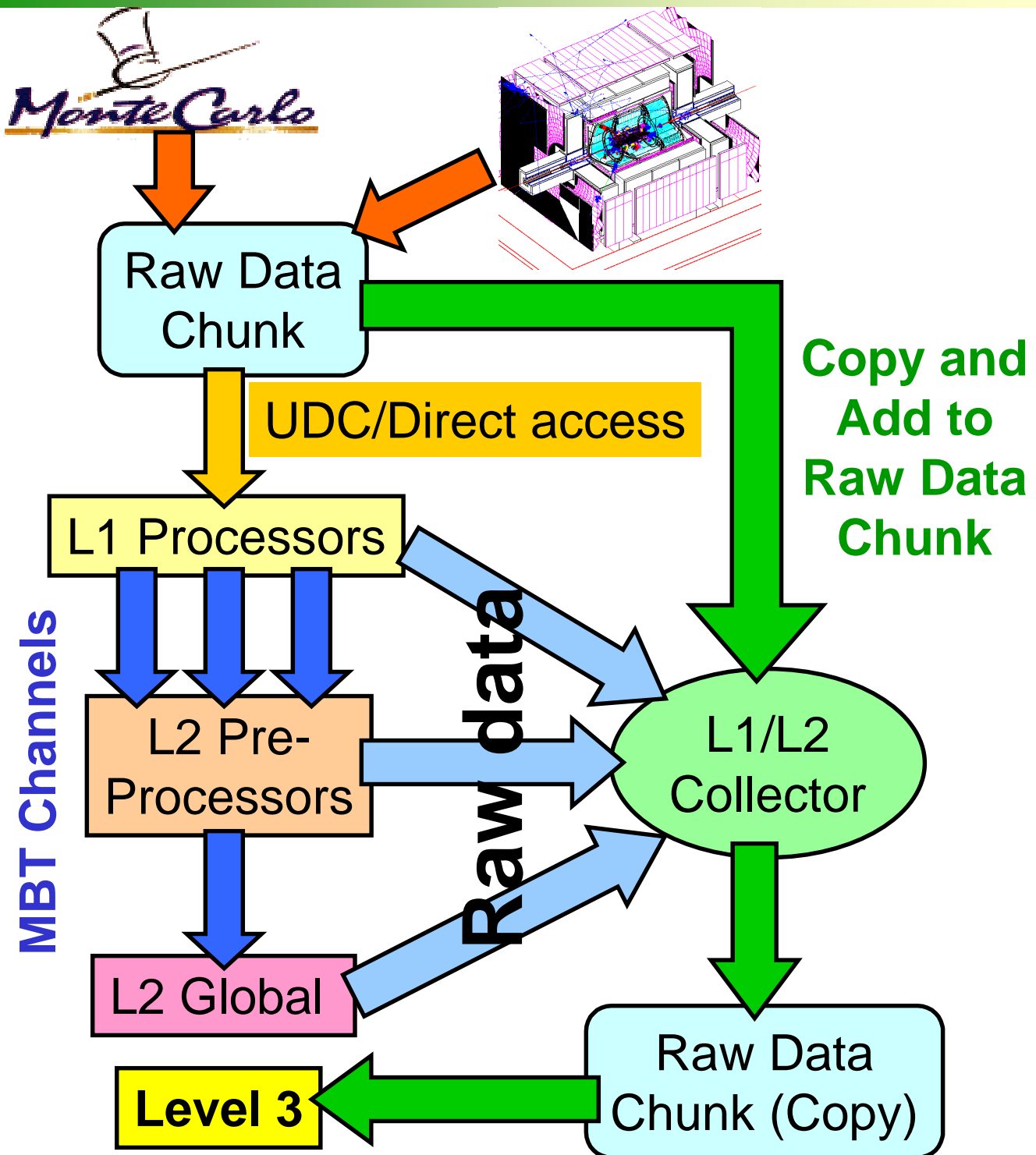
Raw data

L1/L2 Collector

L2 Global

Level 3

Raw Data Chunk (Copy)

Inputs to L1

- L1 inputs from Raw Data Chunk
  - Possibly use Unpacked Data Chunks but not required
  - Alternative route: read Raw Data Chunk directly (I/Ogen)
- L1 framework inputs are specific L1 internal format
- L2 workers will all use I/Ogen to access data
  - Only used in emulation mode w/o L1 running
  - One basic package can work for all "simply" stored inputs

L1 Outputs to L2

- ALL L2 inputs are MBT channels
  - L1 processors are <u>REQUIRED</u> to fill MBT channels to send to L2
  - Format must be same as online L2 inputs

  i.e. L1 processors fulfill their full online functionality
- SLICs/STT very similar to L1 processors
  - Input via Raw/Unpacked DC
  - Output via MBT channel

L1 Example

```
// Example MBT Filler
Result MyClass::ready(DataStore &ds) {
 // Create an MBT channel which can
 // be filled with up to 40 Electrons
 FillableMBTChannel<Electron,40> eleMBT;
 Electron *myele=new Electron;
 // Fill myele with some data
 myele->setPhi(0.1);
 myele->setET(100.);
 …
 // Add myele to the MBT channel
 eleMBT.addObject(myele);
 // Create a data broadcast class
 DataBroadcast *output=
                   new DataBroadcast;
 // Pack the MBT channel into the class
 output.store(eleMBT);
 // Add output to data store
 produceItem(ds,output,"l2output");
 // Return successfully
 return SUCCESS;
}
```

L1/L2 Outputs to L3

- All L1/L2 framework packages produce an "L3Output" object
  - Contains packed, raw data output written to VRB/VBD
- Generated either by
  - I/Ogen (recommended)
    - all L2 uses this method
  - Hand-coded routines
    - for those really strange formats I/Ogen can't handle
- Packing routine should be coded to I/Ogen interface
  - Will make UDC interface easy

L1/L2 Collector

- Configured by a crate readout list from COORsim
- Each incoming raw data packets assigned to a L3 crate
- Crates for new Raw Data Chunk extracted from following sources (in order)
  - L1/L2 Raw Data packets
  - Old Raw Data Chunk
  - New, dynamically allocated, empty crate

L1/L2 Collector

- Has to copy Raw Data Chunk
  - "chunks are immutable"
- Does no packing
  - All input data comes in pre-packed form
- Copying only done when Chunk added to event
  - Chunk built from a system of L3 crate objects
  - Each L3 crate needs a block or raw data (pointer)
  - Automatically handles VBD/VRB headers
  - Thank you Gordon!

L1/L2 Collector

COORsim Crate List

Crate 1	Crate 2	Crate 3	Crate 4	Crate 5

New Raw Data Chunk

Crate 1	Crate 2	Crate 3	Crate 4	Crate 5

L1/L2 Outputs to L3

Crate 1		Crate 3	Crate 4	

Old Raw Data Chunk

Crate 2   Crate 4

Create new, empty crate ⟶ Crate 5
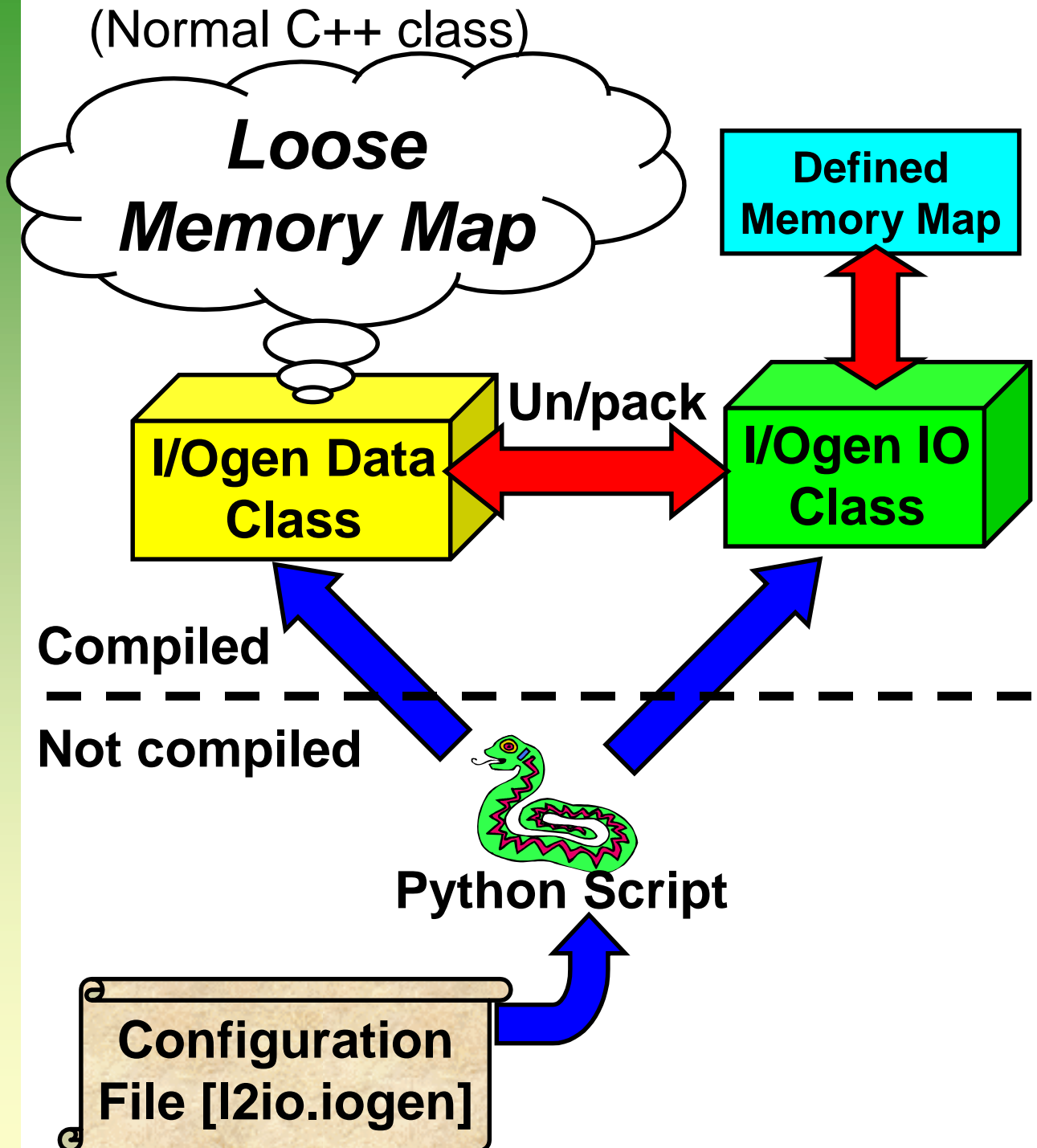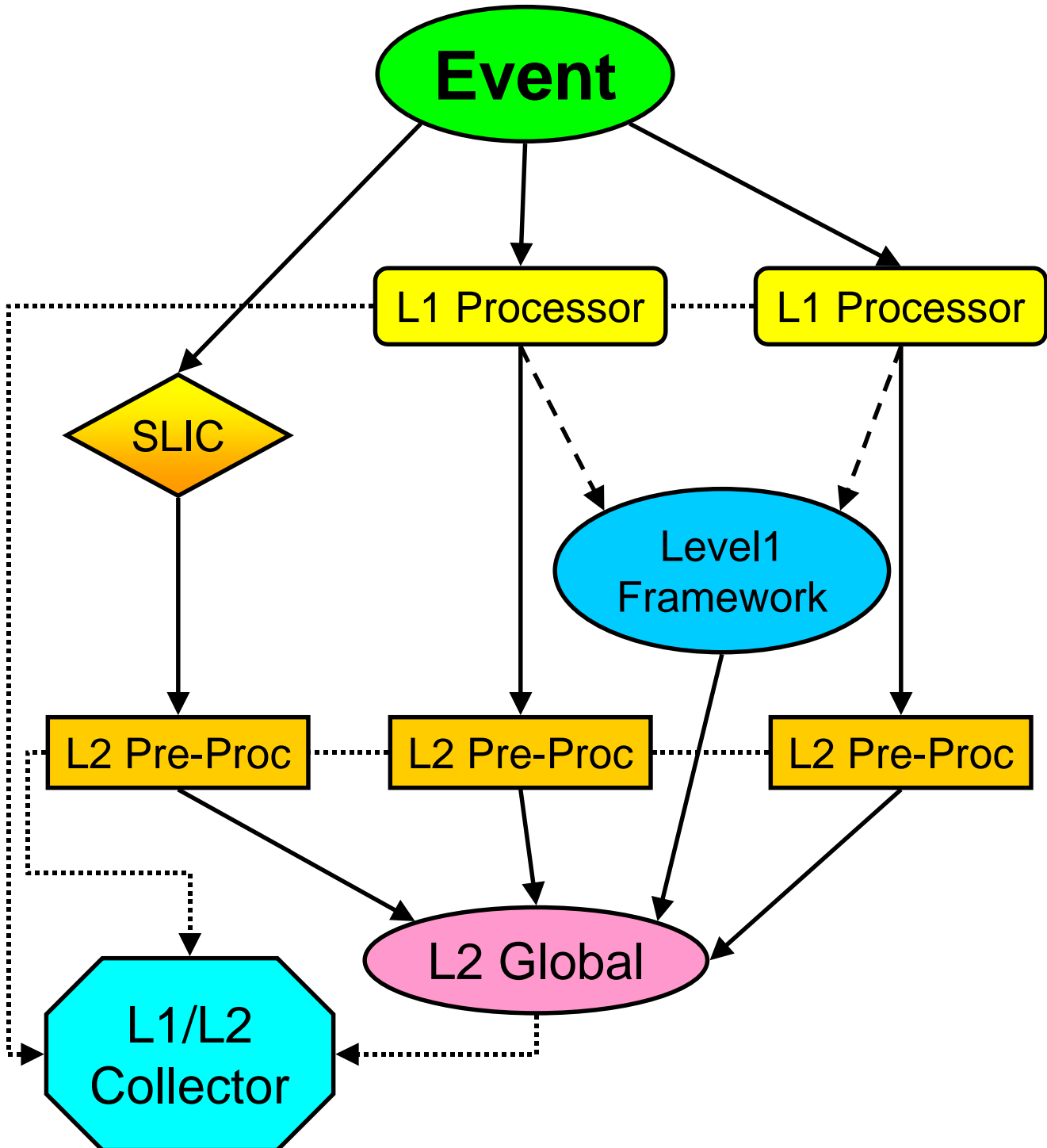
Current Status

- Few days away from working L1/L2 framework
  - Basic functionality only
  - Uses Data Flow framework thanks to Jim Kowalkowski
- Next steps (after release)
  - Insert L1Cal, L2Cal & Global
  - Add L3 output mechanism
  - Add L2Parser Interface to allow fully functional Global

How I/Ogen Works

(Normal C++ class)

Loose Memory Map

Defined Memory Map

I/Ogen Data Class

Un/pack

I/Ogen IO Class

Compiled

Not compiled

Python Script

Configuration File [l2io.iogen]

Offline Design

Data Flow Types

Data Access

Event

L1 Processor

L1 Processor

SLIC

L1 Framework Data

Level1 Framework

L2 Pre-Proc

L2 Pre-Proc

L2 Pre-Proc

L2 Global

MBT Channel